Today: Randomized Algorithms

Next class: Quantum Algorithms

Cool! Neat! Wow!
Randomized Algorithms

Algorithms which uses random bits to solve a problem

Allowed to fail with some probability \( \leq 5\% \)

Sometimes can be much faster than deterministic
Integer factorization

Given a 500 digit number \( N \)
find its prime factorization \( N = p_1 p_2 \ldots p_k \)

Algorithm: Check every integer \( 1 \leq x \leq \sqrt{N} \)
to see if \( x \) divides \( N \)

Runtime:
Suppose \( N \approx 10^{500} \)
Then \( \sqrt{N} \approx \sqrt{10^{500}} = 10^{250} \)

\# of atoms in universe \( \leq 10^{100} \)
\( N \) has \( n \) digits ⇒ needs \( 10^{n/2} \) time

Inefficient!
General number field sieve: Factor an $n$ bit number in time $\approx C n^{1/3} \log(n)^{2/3}$

Factoring is believed to be hard!

But very important:

RSA - 250: 250 digits (factored in 2020)
RSA - 896: 270 digits $\$75,000$
RSA - 2048: 618 digits $\$200,000$

(easy with a quantum computer)
Primality testing

Given an n digit number $N$ determine if it's prime or composite

Idea 1: Factor it! Runs in time $C^{n^{1/3}} \log(n)^{2/3}$

Idea 2: Use something else about prime numbers...

Fermat's little theorem

If $N$ is prime then $a^{N-1} \equiv 1 \pmod{N}$ for all $a \leq 31, \ldots, N-13$
Fermat Test \((N)\)

1. Pick \(a \in \{1, \ldots, N-1\}\) uniformly at random
2. If \(a^{N-1} \equiv 1 \pmod{N}\), output “prime”
   Otherwise, output “composite”

Fact: If \(N\) is prime, always outputs “prime”

\[
N = 12 = 3 \cdot 4 \\
\alpha = 2 \\
2^{11} \not\equiv 1 \pmod{12}
\]

\[
\alpha = 3 \\
3^{11} \not\equiv 1 \pmod{12}
\]

\[
\alpha = 5 \\
5^{11} \equiv 1 \pmod{12}
\]

\(N = p^2\), \(p\) large prime \(\Rightarrow\) only \(\frac{1}{p}\) of \(a\)'s are not coprime

Test only good if \(a^{N-1} \equiv 1 \pmod{N}\) for lots of coprime \(a\)
Carmichael numbers
Composite numbers \( N \) s.t.
\[ a^{N-1} \equiv 1 \pmod{N} \]
for all \( a \) coprime to \( N \)
Pass the Fermat test for all coprime \( a \)
\( N = 561 = 3 \cdot 11 \cdot 17 \)
Let's pretend these don't exist for now...
Thm: Suppose $N$ is composite and not Carmichael. Then $\Pr \left[ \text{Fermat Test}(N) = \text{composite} \right] \geq \frac{1}{2}$

Pf: Not Carmichael $\Rightarrow$ coprime $b$ s.t. $b^{N-1} \neq 1 \pmod{N}$

Claim: Suppose $a$ passes Fermat Test: $(a^{N-1} = 1 \pmod{N})$.
Then $a \cdot b \pmod{N}$ fails Fermat Test.

Pf: $(a \cdot b)^{N-1} = a^{N-1} \cdot b^{N-1} \pmod{N}$
$\equiv b^{N-1} \pmod{N}$
$\neq 1 \pmod{N}$
Need to check
\[ a_i \cdot b \neq a_j \cdot b \pmod{N} \]

**Fact:** \( b \) is coprime

\[ \Rightarrow \text{inverse } b^{-1} \pmod{N} \]

\[ a_i \cdot b \neq a_j \cdot b \pmod{N} \]

\[ b^{-1} \cdot b^{-1} \]

\[ a_i \neq a_j \pmod{N} \]

So \[ |\text{pass}| \leq |\text{fail}| \] \( \square \)
Pr\left[Fermat\; Test\; (N) = \text{composite}\right] \geq \frac{1}{2} \text{ (if } N \text{ is composite)}

Repeat \ k \ times \Rightarrow \text{ detect composite w/ prob } \geq 1 - \frac{1}{2^k}

Can be very confident!
Need to compute \( a^{N-1} \pmod{N} \)

Suppose \( N-1 = 2^n \)

\[
\begin{align*}
    a \cdot a & = a^2 \pmod{N} \\
    a^2 \cdot a^2 & = a^4 \pmod{N} \\
    a^4 \cdot a^4 & = a^8 \pmod{N} \\
    \vdots \\
    a^{2^{n-1}} \cdot a^{2^{n-1}} & = a^{2^n} \pmod{N}
\end{align*}
\]

(Con generalize to arbitrary \( N-1 \))
Can add another check to detect Carmichael numbers
This gives Miller–Rabin primality test (1976)
Since then, we've only had randomized alg's (no deterministic undergrads)

A deterministic alg for primality testing in $O(n^{12})$ time (later $O(n^6)$ time)

Miller 1976 “derandomization”
Try the Miller–Rabin test for all $a \leq O(n^{12})$
This will detect if $N$ is prime or composite (assuming generalized Riemann hypothesis)
Primality: efficient randomized algorithm first, later efficient deterministic alg

Other problems: Only know efficient randomized alg
(Polynomial identity testing)

Two possible worlds: 1. Every efficient randomized alg has deterministic counterpart
2. Some problems only have efficient randomized algs

\[ \text{P} \subset \text{BPP} \subset \text{efficient randomized} \]
Minimum cut

Unweighted, undirected graphs $G = (V, E)$

cut of size 1

cut of size 3
Idea: Max flow/min cut alg

Computes min s-t cut in time $O(n \cdot m)$

Which s,t to use? $\forall \ V_1 \in \mathcal{E}$

Set $s=1$, try all $t=2, \ldots, n$

$O(n^2 \cdot m)$ time
Karger’s algorithm $(G)$

for $i = 1, \ldots, n-2$ \hspace{1cm} (n = |V(G|) \\
1. pick a uniformly random edge $e$ \\
2. contract $e$

return cut specified by the remaining two supervertices

Diagram:

1. $0 \rightarrow 0^2$
2. $3 \rightarrow 4$
3. Contract
4. $2 \rightarrow 3$
5. Contract
6. $1 \rightarrow 2 \rightarrow 3$
7. Contract
Intuition

Karger’s alg finds min cut if it never contracts. But way more edges on left will usually pick there!
**Thm:** Let \( C = (S, \bar{S}) \) be a min cut of size \( k \). 
\[
\Pr\left[\text{Karger's alg outputs } C\right] \geq \frac{1}{\binom{n}{2}^2 n(n-1)} \]

**Pf:** Let \( G_i \) for graph cut beginning of \( i \)th iteration \((G_i = G)\)

**Fact 1:** Min-Cut in \( G_i \) \( \geq k \).
(any cut corresponds to cut in \( G \) )

**Fact 2:** \# of vertices in \( G_i = n - (i-1) \)
\[
= n - it + 1
\]
Fact 3: degree of each vertex in $G_i \geq k$

Fact 4: \# of edges in $G_i$

$$= \frac{1}{2} \sum_{v \in G_i} d(v) \geq \frac{1}{2} \sum_{v \in G_i} k$$

$$= \frac{1}{2} k \cdot |G_0|$$

$$= \frac{1}{2} k \cdot (n-i+1)$$
Suppose at $G_i$, haven't contracted edge in $C$ yet.

$$\Pr \left[ \text{don't contract an edge in } C \right]$$

$$= 1 - \Pr \left[ \text{contract an edge in } C \right]$$

$$\geq 1 - \frac{k}{3 \cdot k \cdot (n-i+1)} = 1 - \frac{2}{(n-i+1)}$$

$$\Pr \left[ \text{never contract edge in } C \right]$$

$$\geq \left(1 - \frac{2}{n}\right) \left(1 - \frac{2}{n-1}\right) \left(1 - \frac{3}{n-2}\right) \cdots \left(1 - \frac{2}{3}\right)$$

$$= \left(\frac{n-2}{n}\right) \left(\frac{n-3}{n-1}\right) \left(\frac{n-4}{n-2}\right) \cdots \left(\frac{2}{4}\right) \left(\frac{1}{2}\right)$$

$$= \frac{2}{n(n-1)}$$
Pr(succeeds) ≥ \frac{1}{\binom{n}{2}} \approx \frac{2}{n^2}

Succeed w/ constant prob: repeat \ n^2 \ times
  (or slightly more)

\# of min cuts ≤ \binom{n}{2}

Pr(output min cut) > \sum_{\text{min cut}} \frac{1}{\binom{n}{2}}